

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

APPLICATION FOR UNITED STATES PATENT

**METHOD AND APPARATUS FOR CREATING A PACKET  
USING A DIGITAL SIGNAL PROCESSOR**

INVENTORS:

Alan DeMars  
4681 Blackstone Court  
Santa Maria, CA 93455  
Citizen of the United States of America

Paul Harding-Jones  
9 Enborne Gardens  
Bracknell  
Berkshire  
RG12 2LF  
Citizen of England

VIA EXPRESS MAIL EK539125257US ON 11/9/2001

## **METHOD AND APPARATUS FOR CREATING A PACKET USING A DIGITAL SIGNAL PROCESSOR**

### **TECHNICAL FIELD OF THE INVENTION**

[0001] The present invention relates generally to the field of communications and, more  
5 particularly, to a method and apparatus for creating a packet using a digital signal processor.

### **BACKGROUND OF THE INVENTION**

[0002] The increasing demand for data communications has fostered the development of  
techniques that provide more cost-effective and efficient means of using communication  
networks to handle more information and new types of information. One such technique is to  
10 segment the information, which may be a voice or data communication, into packets. A  
packet is typically a group of binary digits, including at least data and control information.  
Integrated packet networks (typically fast packet networks) are generally used to carry at  
least two (2) classes of traffic, which may include, for example, continuous bit-rate ("CBR"),  
speech ("Packet Voice"), data ("Framed Data"), image, and so forth. Packet networks  
15 source, sink and/or forward protocol packets. Each packet has a well-defined format and  
consists of one or more packet headers and some data. The header typically contains  
information that gives control and/or address information, such as the source and destination  
of the packet.

[0003] Packet header creation typically requires a significant amount of system resources,  
20 such as a central processing unit ("CPU") where a protocol stack is implemented. As a

result, the throughput of a communications switch is limited or constrained by the capacity of the CPU within the switch and the other processing functions that the CPU must also provide. Such processing constraints cause congestion and Quality of Service ("QoS") problems inside the switch. Accordingly, there is a need for a method and apparatus for creating a packet, including the necessary headers, in a digital signal processor instead of a CPU where a protocol stack is implemented.

### SUMMARY OF THE INVENTION

[0004] The present invention provides a method and apparatus for creating a packet, including the necessary headers, in a digital signal processor instead of a CPU where a protocol stack is implemented. As a result, the present invention relieves a significant portion of system resources, such as the CPU or the router, from the responsibility of header creation. Moreover, the responsibility of header creation is distributed over a number of digital signal processors. The CPU where the protocol stack is implemented is, therefore, free to accomplish other tasks, which necessarily reduces congestion, increases QoS, increases throughput and contributes to the overall system efficiency.

[0005] The present invention provides a method of creating a packet using a digital signal processor. Call set-up information is received. Thereafter, call data is received and a data portion of the packet is created using the call data. One or more headers are created using the call data and the call set-up information. The packet is then created by attaching the one or more headers to the data portion of the packet. In addition, the one or more headers may

include a real time transport protocol header, a user datagram protocol header, an Internet protocol header, and a media access control header.

[0006] In addition, the present invention provides an apparatus containing an array of digital signal processors. Each digital signal processor is programmed to receive call set-up  
5 information, receive call data, create a data portion of the packet using the call data, create one or more headers using the call data and the call set-up information, and create a packet by attaching the one or more headers to the data portion of the packet.

[0007] The present invention also provides a communications switch having one or more cards having ingress, signal processing and egress functions, one or more control cards  
10 containing one or more processors, a switch fabric communicably coupling the one or more cards and the control cards, and a TDM bus communicably coupling the one or more cards and the control cards. The signal processing function of the one or more cards comprises one or more arrays of digital signal processors. Each digital signal processor is programmed to receive call set-up information, receive call data, create a data portion of the packet using the  
15 call data, generate one or more headers using the call data and the call set-up information, and create a packet by attaching the one or more headers to the data portion of the packet.

[0008] Other features and advantages of the present invention shall be apparent to those of ordinary skill in the art upon reference to the following detailed description taken in conjunction with the accompanying drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

[0009] For a better understanding of the invention, and to show by way of example how the same may be carried into effect, reference is now made to the detailed description of the invention along with the accompanying figures in which corresponding numerals in the  
5 different figures refer to corresponding parts and in which:

FIGURE 1 is a block diagram of a representative integrated network in accordance with the prior art;

FIGURE 2A is a schematic diagram of a packet network switch in accordance with the prior art;

10 FIGURE 2B is a flowchart of a header creation method in accordance with the voice gateway in FIGURE 2A;

FIGURE 3A is a schematic diagram of a packet network switch in accordance with the prior art;

15 FIGURE 3B is a flowchart of a header creation method in accordance with the voice gateway in FIGURE 3B;

FIGURE 4 is a diagram of a packet network switch in accordance with the present invention;

FIGURE 5 is a schematic diagram illustrating a packet operating system in accordance with the present invention;

20 FIGURE 6A is a schematic diagram of a packet network switch in accordance with the present invention; and

FIGURE 6B is a flowchart of a header creation method in accordance with the present invention.

#### DETAILED DESCRIPTION OF THE INVENTION

[0010] While the making and using of various embodiments of the present invention are discussed in detail below, it should be appreciated that the present invention provides many applicable inventive concepts, which can be embodied in a wide variety of specific contexts. For example, in addition to telecommunications systems, the present invention may be applicable to other forms of communications or general data processing. Other forms of communications may include communications between networks, communications via satellite, or any form of communications not yet known to man as of the date of the present invention. The specific embodiments discussed herein are merely illustrative of specific ways to make and use the invention and do not limit the scope of the invention.

[0011] The present invention provides a method and apparatus for creating a packet, including the necessary headers, in a digital signal processor instead of a CPU where the protocol stack is implemented. As a result, the present invention relieves a significant portion of system resources, such as the CPU where standard protocol stack operations occur, from the responsibility of header creation. Moreover, the responsibility of header creation is distributed over a number of digital signal processors. The CPU where the protocol stack is implemented is, therefore, free to accomplish other tasks, which necessarily reduces congestion, increases QoS, increases throughput and contributes to the overall system efficiency.

[0012] Now briefly referring to FIGURES 1-3, a representative network (FIGURE 1) and various packet network switches and methods of header creation (FIGURES 2A, 2B, 3A and 3B) will be described in accordance with the prior art. FIGURE 1 depicts a representative integrated network 100 in which phones 102 and faxes 104 are communicably coupled to a public switched telephone network ("PSTN") 106. A switch 108 is communicably coupled to the PSTN 106 and an Internet Protocol ("IP") network 110 to convert time division multiplexing ("TDM") based communications 112 to IP-based communications 114. The switch 108 creates IP packets containing the necessary destination information so that the packets 114 can be properly routed to their destinations, which may include computers 116 or other devices communicably coupled to the IP network 110. A network controller 118 is communicably coupled to the PSTN 106 and the switch 108, and provides control signals to the switch 108 for proper processing of the TDM based communications 112. The network controller 118 may also be communicably connected to the IP network 110. Network controller 118 can function as a Media Gateway Control ("MGC"). The MGC protocol is one of a few proposed control and signal standards to compete with the older H.323 standard for the conversion of audio signals carried on telephone circuits, such as PSTN 106 to data packets carried over the Internet or other packet networks, such as IP network 110. As will be appreciated by those skilled in the art, the present invention is not limited to the conversion of TDM based communications to IP-based communications; instead, the present invention may be applied to any conversion of a multiplexed communication to a packet-based communication.

[0013] IP specifies the format of packets, also called datagrams, and the addressing scheme. IP network 110 receives and sends messages through switch 108, ultimately to phone 102 and/or fax 104. PCs 116 receive and send messages through IP network 110 in a packet-compatible format. Voice over IP (“VoIP”) is the ability to make telephone calls and send faxes over IP-based data networks, such as IP network 110. An integrated voice/data network 100 allows more standardization and reduces total equipment needs. VoIP can support multimedia and multi-service applications.

[0014] Turning now to FIGURES 2A and 2B, a schematic diagram of a packet network switch 200 and a header creation method in accordance with the prior art are shown. As illustrated, the packet network switch 200 includes a digital signal processor (“DSP”) 202 communicably coupled to a CPU 204. The CPU 204 is communicably coupled to a protocol stack 206. During the conversion of a TDM-based communication 112 to an IP-based communication 114, the CPU 204 receives signaling instructions for the call in block 210 and assigns a DSP 202 to process the call in block 212. The DSP 202 receives the call data in block 214. The DSP 202 then compresses the call data and creates a data portion of the packet in block 216. The DSP 202 sends the data portion of the packet to the CPU 204 in block 218. The CPU 204 creates a real time transport protocol (“RTP”) header, attaches the RTP header to the data portion of the packet and sends the packet to the protocol stack 206 in block 220. The protocol stack 206 creates a user datagram protocol (“UDP”) header, Internet protocol (“IP”) header and media access control (“MAC”) header and attaches these headers to the packet in block 222. The protocol stack 206 then sends the complete packet (data plus headers) out over the IP network in block 224. If the call is terminated, as determined in



decision block 226, the call is terminated in block 228. If, however, the call is not terminated, the DSP 202 receives more call data in block 214 and the above-described process repeats until the call is terminated. As illustrated, both the CPU 204 and the protocol stack 206 share the responsibility for header creation in the packet network switch 200.

5 [0015] Signaling instructions refers to the exchange of call control or call set-up information between the switch and other network elements. The purpose of a signaling system is to transfer control information, or signaling units, between elements in a telecommunications system. Early signaling systems carried the control signals on the same circuit as the user traffic. For example, older in-band signaling systems use this approach.

10 Newer signaling systems, such as Common Channel Signaling System 7 ("SS7"), use a separate channel for signaling information. These systems are called common channel signaling systems because a separate (common) channel is used for signaling. Some call this approach "out of band" signaling. Two types of out-of-band signaling exist today. SS7 is an example of the first type, physical out-of-band signaling, in which a separate physical

15 channel is used for signaling. SS7 is usually, but not necessarily, deployed as a separate network within the complete telephone network architecture for the purpose of establishing and terminating telephone calls. SS7 messages are exchanged between network elements over bi-directional channels called signaling links. ISDN is an example of physical in-band, logical out-of-band signaling. In this approach, signaling and user traffic share the same

20 physical transmission medium, but part of the channel capacity is reserved only for signaling traffic. The remainder of the bandwidth is reserved for user traffic, such as the telephone call. Other signaling schemes can be used with T1 and E1 circuits.

[0016] RTP is an Internet standard for the transport of real time data, including audio and video. RTP is used to identify packets as containing a voice sampling in a particular encoding format. A timestamp and sequence number are typically used to reassemble a synchronous voice stream from a stream of RTP packets. RTP can also be used for media on demand services and interactive services like IP telephony. On the other hand, UDP provides efficient but unreliable (nonguaranteed) transport of data. It is used for the transport of real-time voice data since retransmission of real-time data would add too much delay to the voice conversation. IP, however, provides a standard encapsulation of data for transmission over the network. It contains a network-wide source and destination address used for routing. MAC provides the layer two addressing for the local segment of the IP network.

[0017] Turning now to FIGURES 3A and 3B, a schematic diagram of a packet network switch 300 and a header creation method in accordance with the prior art are shown. As illustrated, the packet network switch 300 includes a digital signal processor ("DSP") 302 communicably coupled to a CPU 304. The CPU 304 is communicably coupled to a call control card 306 and a switch fabric 308. During the conversion of a TDM-based communication 112 to an IP-based communication 114, the CPU 304 receives signaling instructions for the call from the call control card 306 in block 310 and assigns a DSP 302 to process the call in block 312. The DSP 302 receives the call data in block 314. The DSP 302 then compresses the call data and creates a data portion of the packet in block 316. The DSP 302 sends the data portion of the packet to the CPU 304 in block 318. The CPU 304 creates a RTP header, a UDP header, an IP header and a MAC header, and attaches these headers to the data portion of the packet in block 320. The CPU 304 then attaches a flow tag

to the packets to bypass the call control card 306 in block 322. The CPU 304 sends the packet to switch fabric 308, which removes the flow tag and sends the complete packet (data plus headers) out over the IP network in block 324. If the call is terminated, as determined in decision block 326, the call is terminated in block 328. If, however, the call is not terminated, the DSP 302 receives more call data in block 314 and the above-described process repeats until the call is terminated. As illustrated, the CPU 304 has sole responsibility for header creation in the packet network switch 300.

**[0018]** Now referring to the present invention and to FIGURE 4, a packet network switch 400 will now be described. The packet network switch 400 can be used to process VoIP, voice over Frame Relay ("VoFR") and other types of calls. Packet network switch 400 includes one or more ingress cards 402a and 402b, one or more signal processing cards 404, one or more control cards 406, one or more egress cards 408a and 408b, a switch fabric 410 and a TDM bus 412. Each signal processing card 404 contains an array of digital signal processors ("DSP") (not shown) and each control card 406 contains one or more processors (not shown). The switch fabric 410 communicably couples the ingress cards 402, the signal processing cards 404, the control cards 406 and the egress cards 408 together. The TDM bus 412 also communicably couples the ingress cards 402, the signal processing cards 404, the control cards 406 and the egress cards 408 together. Preferably cards 402, 404, 406 and 408 can be inserted in any order within packet network switch 400. Moreover, the packet network switch 400 should include sufficient numbers of redundant cards to serve as backup cards in the event a card 402, 404, 406 and 408 fails. Note that the ingress cards 402, signal

processing cards 404, and the egress cards 408 can be replaced by one or more cards having ingress, signal processing and egress functions.

[0019] The main function of the packet network switch 400 shown in FIGURE 4 is to provide media conversion from PCM encoded data to packetized data. When a call or communication is to be handled by the packet network switch 400, a network controller 118 (FIGURE 1) provides the control card 408 with the necessary call set-up information. Control card 408 uses this call set-up information to assign a port in ingress cards 402a or 402b to receive the call from the PSTN 106 (FIGURE 1), a DSP within processing card 404 to process the call, and a port in egress cards 408a or 408b to send the call to IP network 110 (FIGURE 1). The TDM-based communications or messages 112 enter through ingress cards 402a or 402b and are routed to the appropriate processing card 404 through TDM Bus 412. The DSPs in processing card 404 convert messages between analog and digital information formats, and provide digital compression and switching functions. In one embodiment, each processing card 404 is capable of processing 1024 simultaneous sessions. The processing card 404 then sends the messages from the DSP to switch fabric 410, which is primarily responsible for the routing and transferring of messages, the basic transmission unit, between switch elements. The switch fabric 410 may also provide packet buffering, traffic concentration and multiplexing, redundancy for fault tolerance, multicasting or broadcasting, and cell scheduling based on delay priorities and congestion monitoring. Switch fabric 410 ultimately routes the messages to egress cards 408a or 408b. In one embodiment, each egress card 408 is capable of handling at least 8000 calls. Egress cards 408a and 408b

typically send the messages to a gigabit Ethernet (not shown). As its name indicates, the gigabit Ethernet supports data rates of one (1) gigabit (1,000 megabits) per second.

[0020] Turning now to FIGURE 5, a schematic diagram illustrating a packet operating system 500 with redundant control cards 502a and 502b is shown. Control cards 502a and 502b are housed within a single chassis, such as switch 400 (FIGURE 4). Messages 504 enter packet operating system 500 through interface 506 on control card 502a. Messages 504 travel from interface 506 onto protocol stack 508 and then to packet bus 510. Packet bus 510 sends messages 504 to either input/output (“I/O”) cards 512 or DSP cards 514. Control card 502b mirrors either a portion or all of the data of control card 502a. Each control card 502a and 502b of packet operating system 500 has its own memory and thus avoids the typical problems associated with shared memory, such as recursive calls and have synchronization and corruption problems.

[0021] Referring now to FIGURES 6A and 6B, a schematic diagram of a packet network switch 600 and a header creation method in accordance with the present invention are shown.

The packet network switch 600 includes ingress cards 602a and 602b communicably coupled to a TDM bus 604. The TDM bus 604 is communicably coupled to a number of DSPs 606a, 606b, 606c . . . 606n. The DSPs 606a, 606b, 606c . . . 606n are typically configured in an array of DSPs located on one or more signal processing cards 404 (FIGURE 4). Each DSP 606a, 606b, 606c . . . 606n is communicably coupled to a switch fabric 608. The switch fabric is communicably coupled to egress cards 610a and 610b. Packet network switch 600 also includes one or more CPUs 612, which are typically located on one or more control cards 406 (FIGURE 4). The CPU 612 is communicably coupled to the ingress cards 602a

and 602b, the DSPs 606a, 606b, 606c . . . 606n, and the egress cards 610a and 610b. Note that the ingress cards 602a and 602b, the DSPs 606a, 606b, 606c . . . 606n, and the egress cards 610a and 610b can be replaced by one or more cards having ingress, signal processing and egress functions.

5   **[0022]** During the conversion of a TDM-based communication 112 to an IP-based communication 114, the CPU 612 receives signaling instructions 614 for the call in block 620 and assigns an ingress card 602a, 602b port, and egress card 610a, 610b port, and a DSP 606a, 606b, 606c . . . 606n to process the call in block 622. The DSP 606a, 606b, 606c . . . 606n receives call set-up information from the CPU 612 in block 624. Various operating  
10 parameters are typically required to properly configure the DSP 606a, 606b, 606c . . . 606n to process a certain type of call. These parameters can be preset or loaded dynamically using an overlay mechanism. One or more overlays can be stored within the DSP 606a, 606b, 606c . . . 606n or within a separate memory location. For example, the DSP 606a, 606b, 606c . . . 606n can request an overlay from the CPU 612 based on the call set-up information or bearer  
15 type. The DSP 606a, 606b, 606c . . . 606n then receives and loads the overlay. After an overlay has been loaded, DSP 606a, 606b, 606c . . . 606n may perform further discrimination to determine whether a different overlay is required. If the overlay needs to be changed the DSP 606a, 606b, 606c . . . 606n requests a different overlay, and receives and loads the different overlay. For example, the call set-up information may indicate that the bearer type  
20 of the call is voice even though the bearer type may actually be either voice or fax. Thus, if the DSP 606a, 606b, 606c . . . 606n recognizes through further discrimination of the PCM data that the call is actually a fax instead of a voice call, the DSP 606a, 606b, 606c . . . 606n

will request a different overlay so as to properly configure the DSP 606a, 606b, 606c . . . 606n to process the fax.

[0023] Although not required by the present invention, the real-time loading of overlays allows each DSP 606a, 606b, 606c . . . 606n to process any call type. The use of overlays  
5 also allows the packet network switch 600 to be updated to process new call types or more efficiently process existing call types via software updates or downloads. In addition, the packet network switch 600 can use the allocation of overlays to dynamically control the allocation of bandwidth to the various call types to ensure QoS standards and/or compliance with licensing restrictions.

[0024] Next, the DSP 606a, 606b, 606c . . . 606n processes the pulse code modulated  
10 (“PCM”) data in block 626. The DSP 606a, 606b, 606c . . . 606n receives the call data from the assigned ingress card 602a, 602b port via the TDM bus 604 in block 628. The DSP 606a, 606b, 606c . . . 606n then compresses the call data and creates a data portion of the packet in block 630. The DSP 606a, 606b, 606c . . . 606n may also create one or more digital samples  
15 from the compressed call data and create the data portion of the packet using the one or more digital samples. The DSP 606a, 606b, 606c . . . 606n also creates one or more headers, such as a RTP header, a UDP header, an IP header and a MAC header, using the call data and the call set-up information, and attaches these headers to the data portion of the packet in block 632. More specifically, the RTP, UDP and IP header information is provided by the call  
20 control logic at call set-up time and again if they change during the call. The MAC header information is provided from the protocol stack based on routing table and address resolution protocol (“ARP”) table lookups on the IP destination address for the call. The RTP and UDP

headers are determined by the call data while the IP and MAC headers are generated from the call set-up information. Any updates to the MAC header required by a change in network topology are provided from the CPU 612 where the protocol stack is implemented to the DSP 606a, 606b, 606c ... 606n that is generating the complete packet. Such changes are considered to be infrequent. Note that the DSP 606a, 606b, 606c . . . 606n is not limited to the creation of any specific headers, such as a RTP header, a UDP header, an IP header or a MAC header, but can be used to create any header necessary for proper delivery of a packet.

**[0025]** The DSP 606a, 606b, 606c . . . 606n sends the complete packet (data plus headers) to the assigned egress card 610a, 610b port via the switch fabric 608 for transmission out over the IP network in block 634. If the call is terminated, as determined in decision block 636, the call is terminated in block 638. If, however, the call is not terminated, the DSP 606a, 606b, 606c . . . 606n receives more call data in block 628 and the above-described process repeats until the call is terminated. As illustrated, the DSP 606a, 606b, 606c . . . 606n has sole responsibility for header creation in the packet network switch 600. As a result, system resources are conserved because the CPU 612 is relieved from the responsibility of header creation. Moreover, the responsibility of header creation is distributed over a number of DSPs 606a, 606b, 606c . . . 606n. The CPU 612 is, therefore, free to accomplish other tasks, which necessarily reduces congestion, increases QoS, increases throughput and contributes to the overall system efficiency.

**[0026]** The embodiments and examples set forth herein are presented to best explain the present invention and its practical application and to thereby enable those skilled in the art to make and utilize the invention. However, those skilled in the art will recognize that the



foregoing description and examples have been presented for the purpose of illustration and example only. The description as set forth is not intended to be exhaustive or to limit the invention to the precise form disclosed. Many modifications and variations are possible in light of the above teaching without departing from the spirit and scope of the following

5    claims.